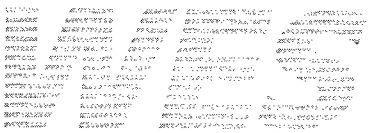
Technical Report ITL-96-7
August 1996

US Army Corps
of Engineers
Waterways Experiment
Station

A Hardware Analysis of the Fundamental Iterative Algorithm for Decoding A (17,9) Binary BCH Code

by Roy L. Campbell, Jr.



Prepared for Headquarters, U.S. Army Corps of Engineers

Technical Report ITL-96-7
August 1996

A Hardware Analysis of the Fundamental Iterative Algorithm for Decoding A (17,9) Binary BCH Code

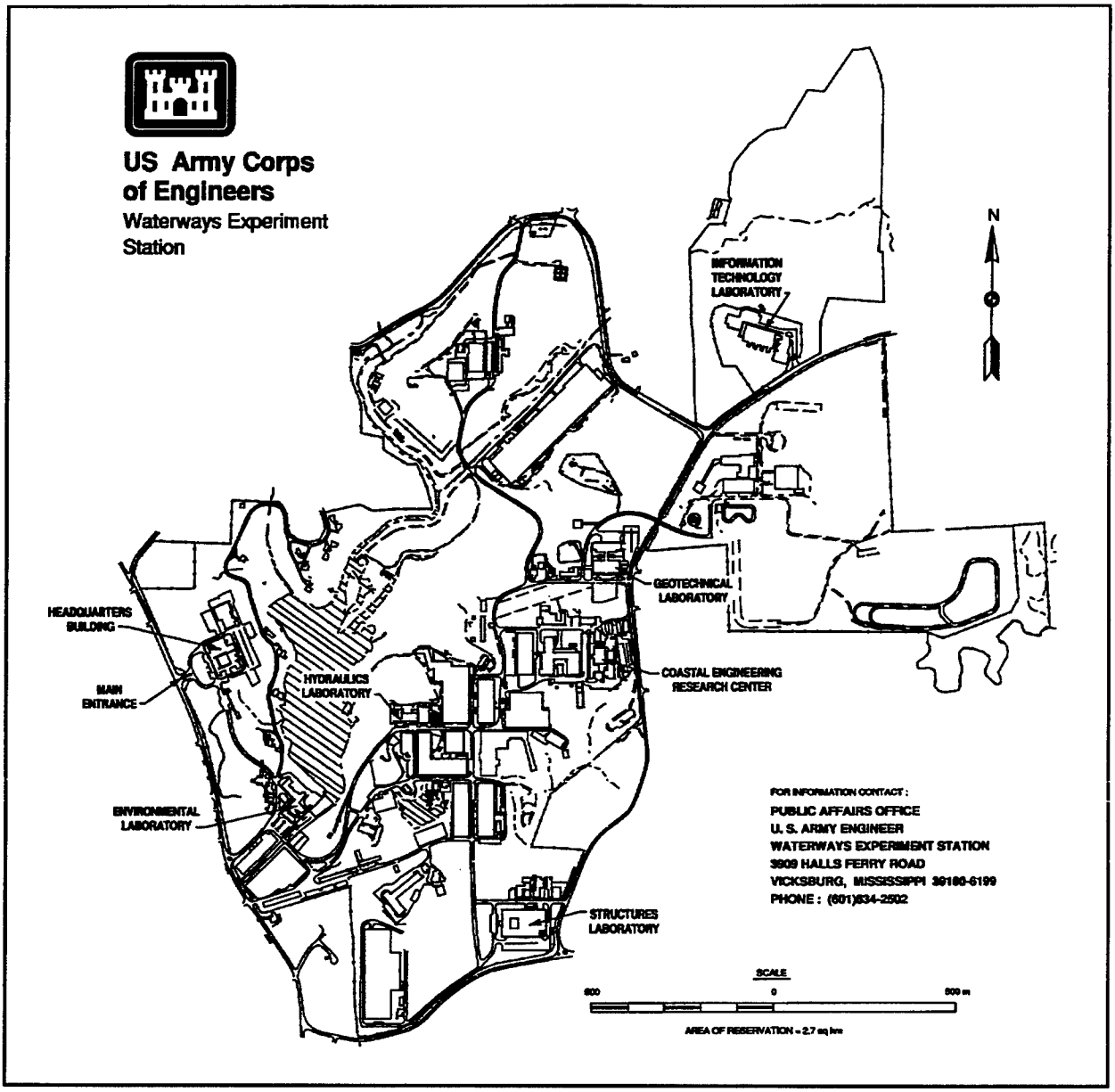
by Roy L. Campbell, Jr.

U.S. Army Corps of Engineers
Waterways Experiment Station
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Final report

Approved for public release; distribution is unlimited

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000



Waterways Experiment Station Cataloging-in-Publication Data

Campbell, Roy L.

A hardware analysis of the fundamental iterative algorithm for decoding a (17,9) binary BCH code / by Roy L. Campbell ; prepared for U.S. Army Corps of Engineers.

44 p. : ill. ; 28 cm. -- (Technical report ; ITL-96-7)

Includes bibliographic references.

1. Computer algorithms. 2. Algorithms. 3. Decoders (Electronics) I. United States. Army. Corps of Engineers. II. U.S. Army Engineer Waterways Experiment Station. III. Information Technology Laboratory (U.S. Army Engineer Waterways Experiment Station) IV. Title. V. Series: Technical report (U.S. Army Engineer Waterways Experiment Station) ; ITL-96-7.

TA7 W34 no.ITL-96-7

PREFACE

This report was prepared by Roy L. Campbell, Jr., Information Management Division (IMD), Information Technology Laboratory (ITL), U.S. Army Engineer Waterways Experiment Station (WES), Vicksburg, Mississippi, as a thesis to the faculty of Mississippi State University in partial fulfillment of the requirements for the Degree of Master of Science in Electrical Engineering in May of 1996. This report was prepared under the supervision of Mr. Murray Huffman, Chief, IMD, and Dr. N. Radhakrishnan, Director, ITL.

At the time of publication of this report, Dr. Robert W. Whalin was Director of WES. COL Bruce K. Howard, EN, was Commander.

ACKNOWLEDGMENTS

Sincere appreciation is extended to Dr. William J. Ebel for his guidance throughout this endeavor, to the Honda Corporation for its financial support of the preceding course work, and to Dr. N. Radhakrishnan, Director of the Information Technology Laboratory, Waterways Experiment Station, for his support and understanding during the thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

A BCH code is a linear cyclic block code with a generator polynomial, $g(x)$, that is a product of minimal polynomials, where the $2t$ designed zeros of $g(x)$ have contiguous powers of β. Note that t is the designed number of correctable errors, $d_o = 2t + 1$ is the designed distance, $\beta = \alpha^b$, α is a primitive element of the field $GF(Q)$, $b = (Q-1)/n$, and n is the length of a channel codeword. Often the true minimum distance of the BCH code is larger than d_o. A decoder based on the Berlekamp-Massey Algorithm (BMA) makes use of the designed distance only [1], however the Fundamental Iterative Algorithm (FIA) can be used to exploit some or all of the code's unused distance, in many cases increasing the number of correctable errors [3].

Often, the critical issue is not the number of correctable errors, but the performance gain/hardware complexity ratio. Since hardware complexity is difficult to measure, this thesis describes and compares hardware realizations for both decoding algorithms in light of their corresponding performance gains.

CHAPTER II

DECODING ALGORITHMS

The Berlekamp-Massey Algorithm

Given a BCH code over $GF(q)$ with a $GF(q^m)$ error locator field, the generator polynomial is $g(x) = LCM[m_1(x), m_2(x), ..., m_{2t}(x)]$, where $m_i(x)$ is the minimal polynomial for β^{j_o+i-1}, for $i = 1, 2, ..., 2t$. Note that $0 \leq j_o \leq q^m - 2$, $\beta = \alpha^b$, α is a primitive element of $GF(q^m)$, $b = (q^m - 1)/n$, and n is the number of code symbols in a codeword. Considering the k coefficients of $i(x)$ as an information block, a channel codeword is obtained by polynomial multiplication

$$c(x) = i(x)g(x),$$

where $\deg\{i(x)\} = k - 1$, $\deg\{g(x)\} = n - k$, $\deg\{c(x)\} = n - 1$, and $k < n$.

The channel errors can be expressed as

$$e(x) = \sum_{j=1}^{\nu} e_{l_j} \cdot x^{l_j},$$

where ν is the number of errors, l_j is the j^{th} error position, and e_{l_j} is the corresponding error weight. Assuming the n symbols of the received codeword are the coefficients of $v(x)$,

$$v(x) = c(x) + e(x),$$

since the channel errors are modeled to be additive. The syndrome values can be calculated as

$$S_i = v(\beta^{j_o+i-1}) = c(\beta^{j_o+i-1}) + e(\beta^{j_o+i-1}) = e(\beta^{j_o+i-1}),$$

for $i = 1, 2, ..., 2t$. By defining $Y_j = e_{l_j}$ and $X_j = \beta^{l_j}$ for $j = 1, 2, ..., \nu$, the syndrome equation can be rewritten as

$$S_i = \sum_{j=1}^{v} Y_j \cdot X_j^{(j_o+i-1)}.$$

for $i = 1, 2, ..., 2t$. Therefore, there are $2t$ non-linear simultaneous equations that relate the channel errors to the syndromes. An error locator polynomial has roots X_j^{-1}, for $j = 1, 2, ..., \nu$, and can be used with a little deduction to reveal the error-syndrome relationship

$$\Lambda(x) = \Lambda_\nu \cdot x^\nu + \Lambda_{\nu-1} \cdot x^{\nu-1} + ... + \Lambda_1 \cdot x + 1 = (1 - x \cdot X_1)(1 - x \cdot X_2)...(1 - x \cdot X_\nu),$$

$$Y_j X_j^{i+\nu} \cdot \Lambda(X_j^{-1}) = Y_j X_j^{i+\nu} \cdot (\Lambda_\nu \cdot X_j^{-\nu} + \Lambda_{\nu-1} \cdot X_j^{-\nu+1} + ... + \Lambda_1 \cdot X_j^{-1} + 1) = 0,$$

$$Y_j \cdot (\Lambda_\nu \cdot X_j^i + \Lambda_{\nu-1} \cdot X_j^{i+1} + ... + \Lambda_1 \cdot X_j^{\nu+i-1} + X_j^{\nu+i}) = 0,$$

$$\sum_{j=1}^{\nu} Y_j \cdot (\Lambda_\nu \cdot X_j^i + \Lambda_{\nu-1} \cdot X_j^{i+1} + ... + \Lambda_1 \cdot X_j^{\nu+i-1} + X_j^{\nu+i}) = 0,$$

$$\Lambda_\nu \sum_{j=1}^{\nu} Y_j X_j^i + \Lambda_{\nu-1} \sum_{j=1}^{\nu} Y_j X_j^{i+1} + ... + \Lambda_1 \sum_{j=1}^{\nu} Y_j X_j^{\nu+i-1} + \sum_{j=1}^{\nu} Y_j X_j^{\nu+i} = 0.$$

Assuming $j_o = 1$,

$$\Lambda_\nu \cdot S_i + \Lambda_{\nu-1} \cdot S_{i+1} + ... + \Lambda_1 \cdot S_{\nu+i-1} + S_{\nu+i} = 0,$$

$$\Lambda_1 \cdot S_{\nu+i-1} + ... + \Lambda_{\nu-1} \cdot S_{i+1} + \Lambda_\nu \cdot S_i = -S_{\nu+i},$$

for $i = 1, 2, ..., 2t$. All $2t$ equations can be expressed at once with the following matrix equation

$$\begin{bmatrix} S_1 & S_2 & ... & S_{\nu-1} & S_\nu \\ S_2 & S_3 & ... & S_\nu & S_{\nu+1} \\ . & & & & . \\ . & & & & . \\ S_\nu & S_{\nu+1} & ... & S_{2\nu-2} & S_{2\nu-1} \end{bmatrix} \cdot \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ . \\ . \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{\nu+1} \\ -S_{\nu+2} \\ . \\ . \\ -S_{2\nu} \end{bmatrix}.$$

Assuming that $\nu \leq t$, the BMA can be used to find the coefficients of the error locator polynomial $\Lambda(x)$ [1]. The initialization of the BMA depends on the first syndrome. If $S_1 \neq 0$, $\Lambda^{(0)}(x) = 1$, $\Lambda^{(1)}(x) = 1 - S_1 x$, $L = 1$, $m = 1$, $\Delta m = S_1$, and $r = 2$; otherwise, $\Lambda^{(-1)}(x) = \Lambda^{(0)}(x) = \Lambda^{(1)}(x) = 1$, $L = 0$, $m = 0$, $\Delta m = 1$, and $r = 2$. $\Lambda^{(r)}(x)$ is the current approximation of the error locator polynomial, L is the stored register length, m is the stored stage number, Δm is the stored syndrome error, r is the current stage number, S'_r is the approximation of the r^{th} syndrome, and Δr is the current syndrome error. The following pseudo code describes the inductive loop of the BMA:

while $(r \leq 2t)$

{

$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x)$

$S'_r = -\sum_{j=1}^{n-1} \Lambda_j^{(r-1)} S_{r-j}$

$\Delta r = S_r - S'_r$

$if\ (\Delta r \neq 0)$

{

$A = -\frac{\Delta r}{\Delta m}$

$l = r - m$

$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) + Ax^l \Lambda^{(m-1)}(x)$

$if\ (\deg\{\Lambda^{(r)}(x)\} > L)$

{

$L = \deg\{\Lambda^{(r)}(x)\}$

$m = r$

$\Delta m = \Delta r$

}

}

$r = r + 1$

}.

$\Lambda^{(2t)}(x)$ is the error locator polynomial $\Lambda(x)$, and L is the number of received symbol errors ν. If $\deg\{\Lambda(x)\} \neq L$, a decoding failure has occurred [1].

A Chien search is an evaluation of a polynomial at all possible values of β^x, where x is arbitrary. This search can be used to find the ν zeros of $\Lambda(x)$. The Forney Algorithm [1] can then be used to find the error weights

$$Y_j = -\frac{\Omega(\beta^{-l_j})}{\Lambda'(\beta^{-l_j})},$$

where

$$\begin{aligned}
\Omega(x) &= S(x)\Lambda(x)(\mathrm{mod}\, x^{2t}), \\
S(x) &= \sum_{j=1}^{2t} S_j x^{(j-1)}, \\
\Lambda'(x) &= \sum_{i=1}^{\nu} \left[\sum_{j=1}^{i} 1 \right] \Lambda_i x^{(i-1)}.
\end{aligned}$$

Now, $e(x)$ is fully specified and

$$\begin{aligned}
c(x) &= v(x) - e(x), \\
i(x) &= c(x)/g(x).
\end{aligned}$$

For the $(17, 9)$ BCH code, the error locator field is $GF(2^8)$, $b = 15$, $\beta = \alpha^{15}$, $g(x) = (x-\beta)\cdot(x-\beta^2)\cdot(x-\beta^4)\cdot(x-\beta^8)\cdot(x-\beta^9)\cdot(x-\beta^{13})\cdot(x-\beta^{15})\cdot(x-\beta^{16})$, and the syndromes are calculated as follows, assuming $j_o = 1$,

$$\begin{aligned}
S_1 &= v(\beta), \\
S_2 &= v(\beta^2) = (S_1)^2.
\end{aligned}$$

This last equation can be proven by considering a generic polynomial defined over $GF(q)$, where q is the characteristic of the field $GF(Q)$,

$$f(x) = \sum_{i=0}^{n-1} f_i \cdot x^i.$$

If the argument of $f(x)$ is raised to the q power,

$$f(x^q) = \sum_{i=0}^{n-1} f_i \cdot x^{iq}.$$

Since $f_i \in GF(q)$, $f_i = f_i^q$. Therefore [1],

$$f(x^q) = \sum_{i=0}^{n-1} f_i^q \cdot x^{iq} = \left[\sum_{i=0}^{n-1} f_i \cdot x^i\right]^q = [f(x)]^q.$$

For all single-error correcting binary BCH codes, the assumption that $j_o = 1$ forces the Berlekamp-Massey Algorithm to degenerate into a mere comparison. If $S_1 = 0$, then $\Lambda(x) = 1$ and $L = 0$. Therefore, the received codeword has no errors, or a decoding error has occurred. In this case, the received codeword is passed as the decoded codeword. If $S_1 \neq 0$, then $\Lambda(x) = 1 - S_1 \cdot x$ and $L = 1$. Therefore, the received codeword has a single error, or a decoding error has occurred. For this case, the zero of the error locator polynomial is extracted by simply noting that for $\Lambda(x) = 0$, $x = 1/S_1$. Next, the error location is given by $l_1 = \log_\beta(S_1)$, and the l_1 bit of the received codeword is complemented. These two cases constitute all possibilities since $S_2 = (S_1)^2$. Therefore, decoding failures are impossible.

The Fundamental Iterative Algorithm

The FIA uses some or all the zeros of $g(x)$ that the BMA does not, to try to increase the decoded distance of the code. Given the same received block $v(x)$ discussed for the BMA, the syndromes for the matrix Ξ can be calculated such that $S_i = v(\beta^i)$, where

$$\Xi = \begin{bmatrix} S_a & S_{a+i_1c_1} & \cdots & S_{a+i_{t'}c_1} \\ S_{a+c_2} & S_{a+i_1c_1+c_2} & \cdots & S_{a+i_{t'}c_1+c_2} \\ \cdot & \cdot & \cdots & \cdot \\ S_{a+(t'-1)c_2} & S_{a+i_1c_1+(t'-1)c_2} & \cdots & S_{a+i_{t'}c_1+(t'-1)c_2} \end{bmatrix}.$$

Note that t' is the FIA's number of correctable errors, as opposed to t, which is the BMA's number of correctable errors. The variables a, c_1, c_2, i_1, i_2, ... , $i_{t'}$ are code-specific and have permanent values for each code, where $0 < i_1 < ... < i_{t'}$. Maximizing t', these permanent values are deduced by trial and error using the whole set or a subset of the zeros of $g(x)$ [3].

Since Ξ possesses recurrent rows (i.e. rows equally spaced by an index of c_2), the decoder algorithm complexity is $O(n^2)$. With $c_1 = c_2 = 1$ and $i_{j+1} = i_j + 1$, for all $j \in 0, ..., t' - 1$, where $i_0 = 0$, the FIA degenerates into the BMA since the entries of Ξ constitute a single set of $2t$ known consecutive syndromes [3].

Given Ξ, if the number of symbol errors in the received codeword is less than or equal to t', then the FIA can be used to find the coefficient polynomial $\psi(x)$ and the rank of Ξ, λ. If the FIA degenerates into the BMA, $\psi(x) = \Lambda(x)$. Otherwise, the relationship of $\psi(x)$ and $\Lambda(x)$ is not as clear. In general,

$$f(x) = \sum_{j=0}^{\lambda} \psi_j \cdot x^{i_{\lambda-j}},$$

where $i_0 = 0$. Also,

$$f(x) = h(x) \cdot \Lambda(1/x) \ ,$$

where $h(x) \in GF(q^m)[x]$. Therefore, $\psi(x)$ and $\Lambda(x)$ have a relationship that is clouded by $h(x)$, which is codeword-dependent [3].

To initialize the FIA, the discrepancy list is set to zero ($D = [0, 0, ..., 0]$), the discrepancy row position list is set to zero ($u = [0, 0, ..., 0]$), the column position is set to 1 ($s = 1$), the row position is set to 1 ($r = 1$), and $\psi(x) = 1$. The following pseudo code describes the inductive section of the FIA, where $d_{r,s}$ is the discrepancy for the current row and column and $\varphi^{(r)}(x) = 1 + \sum_{i=1}^{t'+1} \Xi_{r,i} \cdot x^i$ [3],[4]:

label 1:

$d_{r,s} = [\psi(x) \cdot \varphi^{(r)}(x)]_s = \sum_{i=0}^{s-1} \psi_i \cdot \Xi_{r,s-i}$

label 2:

```
if (d_{r,s} = 0)
        {
        if (r = t')
                {
                λ = s − 1
                END
                }
        else
                {
                r = r + 1
                goto label 1
                }
        }
else
```

{
if $(D_r = 0)$
{
$D_r = d_{r,s}$
$u_r = s$
$\psi^{(s)}(x) = \psi(x)$
$s = s + 1$
$r = 1$
goto label 1
}
else
{
$\psi(x) = \psi(x) - \frac{d_{r,s}}{D_r} \cdot \psi^{(u_r)}(x) \cdot x^{s-u_r}$
$d_{r,s} = 0$
goto label 2
}
}.

With $\psi(x)$ and λ known, the syndrome dependence polynomial $f(x)$ can be constructed. Since $\Lambda(1/x)$ has zeros of the form β^{e_l}, where e_l is the position of the l^{th} potential error, a Chien search can be used on $f(x)$ to eliminate as many zeros of $h(x)$ as possible. All the zeros found with the Chien search are considered members of the set U, where $|U| = \delta$. Using these members, a polynomial similar to $\Lambda(x)$ can be constructed

$$\Gamma(x) = \prod_{l=1}^{\delta}(1 - x \cdot U_l).$$

Some zeros of $h(x)$ might be used in building $\Gamma(x)$. Otherwise, $\Gamma(x) = \Lambda(x)$, which is

capable of revealing t' or less error locations, as opposed to the BMA's $\Lambda(x)$, which can only reveal t or less error locations [3].

Given the largest set of known consecutive syndromes ξ, the Forney Algorithm requires $|\xi| \geq \delta$ or more specifically $|\xi| = \max\{\delta\}$. For all cases where Ξ has entries that form two or more distinct sequences of known consecutive syndromes (i.e. multi-sequence cases), $|\xi| \leq 2t' - 1$. Therefore, if $\delta \geq 2t'$, the number of known consecutive syndromes must be increased. These extra syndromes may or may not be found using the n syndrome linear dependence equations or "patch" equations described by

$$S_{a+i_\lambda c_1+jc_2} + f_1 S_{a+i_{\lambda-1}c_1+jc_2} + ... + f_\lambda S_{a+jc_2} = 0,$$

where $j \in 0, ..., n-1$. Only t' of these equations are necessary, one for each possible non-zero value of λ. The value of j for each equation can be found by trial and error. If the extra syndromes cannot be found, the FIA will not be able to correct t' errors even though Ξ is a $t' \times t'+1$ matrix of the form described previously [3].

Assuming $|\xi| \geq \delta$, an adapted form of the Forney Algorithm can be used to find the error weights

$$W_{e_j} = -\frac{\Omega(\beta^{-e_j})}{\Gamma'(\beta^{-e_j})},$$

where

$$\begin{aligned} \Omega(x) &= S(x)\Gamma(x)(\mathrm{mod}\, x^{|\xi|}), \\ S(x) &= \sum_{j=1}^{|\xi|} S_{a+j-1}x^{(j-1)}, \end{aligned}$$

$$\Gamma'(x) = \sum_{i=1}^{\delta} \left[\sum_{j=1}^{i} 1 \right] \Gamma_i x^{(i-1)}.$$

Therefore, $e(x)$ is fully specified by

$$e(x) = \sum_{j=1}^{\delta} W_{e_j} \cdot x^{e_j},$$

and,

$$c(x) = v(x) - e(x),$$

$$i(x) = c(x)/g(x).$$

For the $(17, 9)$ code, $t' = 2$. The syndrome matrix Ξ is

$$\Xi = \begin{bmatrix} S_1 & S_8 & S_{15} \\ S_2 & S_9 & S_{16} \end{bmatrix},$$

where $a = 1$, $c_1 = 1$, $c_2 = 1$, $i_1 = 7$, and $i_2 = 14$. Since $\delta \leq 8$, $|\xi|$ needs to be 8, so the following patch equations were found by trial-and-error by evaluating the linear dependence equation at $j = 0, ..., n-1$:

$$S_{15} \cdot \psi_1 + S_5 = 0, \text{ for } \lambda = 1 \text{ and } j = 14,$$

$$S_8 \cdot \psi_2 + S_{15} \cdot \psi_1 + S_5 = 0, \text{ for } \lambda = 2 \text{ and } j = 7.$$

For $j = t', ..., n-1$, the equation does not contain syndromes from a particular row in Ξ. These syndromes are contained in a particular extension row of Ξ. For example, S_8, S_{15}, and S_5 comprise the sixth extension row of Ξ, which corresponds to $j = 7$.

For $j = 14$, S_{15} and S_5 comprise the first two elements of the thirteenth extension row. Note that S_1, S_2, S_4, S_8, S_9, S_{13}, S_{15}, and S_{16} are all known. If S_3, S_5, S_6, and S_7, could be found, $|\xi| = 8$. Therefore, for each possible value of λ, the syndrome dependence equation was examined for each value of j, until it became an equation with a single-term unknown that was in the conjugate set containing S_3, S_5, S_6, S_7, S_{10}, S_{11}, S_{12}, and S_{14}.

The generator polynomial is the same as the that for the BMA, but the following version is multiplied out to show the coefficients in $GF(2)$

$$g(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1.$$

As an example, assume that the information polynomial is $i(x) = 1$ and the error polynomial is $e(x) = x^8 + 1$. Then, the codeword polynomial is $c(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1$, and the received polynomial is $v(x) = x^7 + x^6 + x^4 + x^2 + x$. Calculating syndrome S_1 and using the relationships $S_2 = (S_1)^2$, $S_8 = (S_2)^4$, $S_{16} = (S_8)^2$, $S_{15} = (S_{16})^2$, and $S_9 = (S_{15})^4$ gives

$$\Xi = \begin{bmatrix} \alpha^9 & \alpha^{72} & \alpha^{33} \\ \alpha^{18} & \alpha^{132} & \alpha^{144} \end{bmatrix}.$$

Now, $\psi(x)$ and λ can be found with the FIA:

Initialization:

$$D = [0,0]; u = [0,0]; s = 1; r = 1; \psi(x) = 1$$

Loop 1:

$$d_{1,1} = \psi_0 \cdot \Xi_{1,1} = \alpha^9$$

$$D = [\alpha^9, 0]; u = [1, 0]; \psi^{(1)}(x) = 1; s = 2; r = 1$$

Loop 2:

$$d_{1,2} = \psi_0 \cdot \Xi_{1,2} + \psi_1 \cdot \Xi_{1,1} = \alpha^{72}$$

$$\psi(x) = \psi(x) - \frac{d_{1,2}}{D_1} \cdot \psi^{(1)}(x) \cdot x = 1 + \alpha^{63}x; r = 2$$

Loop 3:

$$d_{2,2} = \psi_0 \cdot \Xi_{2,2} + \psi_1 \cdot \Xi_{2,1} = \alpha^{64}$$

$$D = [\alpha^9, \alpha^{64}]; u = [1, 2]; \psi^{(2)}(x) = 1 + \alpha^{63}x; s = 3; r = 1$$

Loop 4:

$$d_{1,3} = \psi_0 \cdot \Xi_{1,3} + \psi_1 \cdot \Xi_{1,2} + \psi_2 \cdot \Xi_{1,1} = \alpha^{254}$$

$$\psi(x) = \psi(x) - \frac{d_{1,3}}{D_1} \cdot \psi^{(1)}(x) \cdot x^2 = 1 + \alpha^{63}x + \alpha^{245}x^2; r = 2$$

Loop 5:

$$d_{2,3} = \psi_0 \cdot \Xi_{2,3} + \psi_1 \cdot \Xi_{2,2} + \psi_2 \cdot \Xi_{2,1} = \alpha^{161}$$

$$\psi(x) = \psi(x) - \frac{d_{2,3}}{D_2} \cdot \psi^{(2)}(x) \cdot x = 1 + \alpha^{199}x + \alpha^{75}x^2$$

$$\lambda = 2$$

This $\psi(x)$ and λ yield $f(x) = x^{14} + \alpha^{199}x^7 + \alpha^{75}$. Using a Chien search, the zeros of $f(x)$ are $\alpha^{0 \cdot 15}$ and $\alpha^{8 \cdot 15}$; therefore, $\Gamma(x) = \alpha^{120}x^2 + \alpha^9 x + 1$. For this code, $|\xi| = 8$. Also, for this example, $\lambda = 2$. Therefore, S_3, S_5, S_6, and S_7 must be determined using the $\lambda = 2$ patch equation in order to have 8 known consecutive syndromes. All

four unknown syndromes are in the same conjugate set, so S_5 can be deduced from the patch equation and then the following equations can be used to find S_3, S_6, and S_7:

$$\begin{aligned} (S_5)^4 &= S_3, \\ (S_3)^2 &= S_6, \\ (S_6)^4 &= S_7. \end{aligned}$$

It may be confusing why $|\xi| = \max\{\delta\} = 8$. This value is determined by constructing a FIA decoder in software. This software is used to decode all possible error patterns containing t' or less errors starting with $|\xi| = 2t'$ and increasing $|\xi|$ after each attempted verification until the decoder corrects all of these error patterns. This process finds the minimum value of $|\xi|$ required to properly decode t' or less errors.

With S_1 through S_8 and $\Gamma(x)$ known, the Forney algorithm can be used to identify which zeros of $f(x)$ of the form β^x correspond to actual error locations. For this example, both zeros are identified, but there are many cases where the identified zeros are a subset of the zeros found with the Chien search. With the two zeros identified, $e(x) = x^8 + 1$, which is the correct error pattern.

The relation between the FIA and the BMA is not easily seen; however, there is one. Starting with column one, the FIA uses a truncated BMA; it starts out like the BMA but stops if a discrepancy, $d_{r,s}$, is non-zero. This discrepancy and its column number are stored as D_r and u_r along with the current approximation of the coefficient

polynomial, $\psi(x)$, which is stored as $\psi^{(s)}(x)$. For the other columns, the initialization is dependent upon all previous columns; the procedure is similar to the procedure for the first column, except when a non-zero discrepancy is encountered, the truncated BMA will stop only if no discrepancy has been recorded for that particular row, in essence, if $D_r = 0$. If $d_{r,s} \neq 0$ and $D_r \neq 0$, $\psi(x)$ is updated. The algorithm continues this pattern for each column until it is able to perform the full BMA on a column. It is possible that the full BMA might be performed on the first column, but this would indicate that no errors could be found.

CHAPTER III

PERFORMANCE COMPARISONS

Since theoretical performance measures of BCH codes are inexact, simulation was necessary to empirically measure performance. Therefore, a baseband digital communication system using BMA and FIA decoders was programmed on a computer. This simulated system was used to approximate the bit error rate (BER) at the decoder output versus SNR curve for both the BMA and the FIA. This curve is important since it gives insight into the system coding gain, which is an important performance indicator.

Let $p_u(h)$ be the BER for the uncoded system operating at the SNR h, and let $p_c(h)$ be defined in a similar way for the coded system. Then, the coding gain is defined as

$$g(\rho) = p_c^{-1}(\rho) - p_u^{-1}(\rho),$$

where ρ is the BER and all SNR's are in dB. Therefore, the coding gain at a given BER is the difference in SNR of the uncoded and coded systems when each are operating at a specific BER [11].

The Simulated System

This statistical computer simulation entails:

1. Generating an X bit maximum entropy encoder input sequence,
2. Encoding in k symbol blocks at a k/n input/output rate, which involves constructing an n-length channel codeword from a k-length source codeword, where each component is a binary symbol,
3. Generating an effective channel error sequence of length Xn/k such that the errors are exponentially spaced. The probability of an error is approximately the BER of the coded channel, and the number of decoder bit errors χ is large (greater than 100) [2],
4. Adding the error sequence to the encoded sequence bit by bit in $GF(2)$,
5. Decoding in n symbol blocks at an n/k input/output rate.

It is assumed that source encoding, if necessary, has been applied so that the binary symbols at the encoder input are independent and equally likely. This input can be modeled by a pseudo-noise (PN) sequence. A PN sequence is periodic with a maximum period of $N = 2^r - 1$, where r is the number of stages in the generating shift register [10]. In many cases, tap weights can be selected so that the maximum period is achieved, resulting in a maximal-length sequence [10]. Each maximal-length sequence has at least three non-zero taps. Those sequences with only three taps are the most time efficient, since the generation speed increases as the number of non-zero taps decrease. The periodic autocorrelation $R_p(u)$ is defined as $(N_0 - N_1)/(N_0 + N_1)$,

where p is a sequence with period $N = N_0 + N_1$, s is the modulo-2 addition of p and its u position cyclic shift, N_0 is the number of 0's in s, and N_1 is the number of 1's in s. Any modulo-2 addition of a maximal-length sequence with its u position cyclic shift is the u' position cyclic shift of the maximal-length sequence. Also, the number of 1's in a maximal-length sequence is one greater than the number of 0's; therefore, the periodic autocorrelation function of any maximal-length sequence is [10]

$$R_p(u) = \begin{cases} 1.0 & \text{if } u = lN \\ -\frac{1}{N} & \text{if } u \neq lN \end{cases}$$

where $l \in I$. As r approaches ∞, N approaches ∞, and the correlation dwindles to 0.

For a simulation, $N = 2^x - 1$, where $x = \lceil log_2 X \rceil$ and X is the number of input bits required by the encoder for an entire simulation. X is difficult to determine, since the calculation of X is based on the coded BER ρ

$$X = \frac{(\chi)(k)}{\rho n},$$

where χ is the number of bit errors at the output of the decoder. The BER at the decoder output is actually a random variable (RV), which can have a multimodal distribution if X is not large enough. The goal is to choose X so that χ is large (greater than 100) [2].

The probability of a bit error at the output of the raw channel when an encoder and decoder are present is the BER of the coded channel ψ_0. If $\psi_0 \geq 10^{-2}$, the effective

channel error sequence can be constructed by thresholding successive outputs of a uniform pseudo random number with ψ_0. If the output is less than or equal to ψ_0, a 1 is declared; otherwise, a 0 is declared. If $\psi_0 < 10^{-2}$, the following approach is used. The location of the first error is calculated by [6]

$$\Delta s = -\frac{1}{\psi_o} \ln(u),$$

where Δs is the number of zero sequence elements until an error occurs and u is a uniform pseudo random number. Subsequent error locations can be determined by using the same equation where Δs is the number of zero sequence elements until the next error and u is a new uniform pseudo random number [6].

The Results

For the $(17, 9)$ BCH code with $g(x) = m_1(x)$ and a minimum distance $d = 5$, the exploited distance is 4 for the BMA and 5 for the FIA, yielding $t = 1$ and $t' = 2$, respectively. Figure 3.1 shows the corresponding theoretical performance curves of error probability versus E_b/N_o. The theoretical BER at the output of the decoder is given by [7]

$$P(\text{bit error at decoder output}) = \sum_{j=t+1}^{n} \left[\frac{j}{n} \cdot \binom{n}{j} \cdot p^j \cdot (1-p)^{n-j} \right],$$

where

$$P(j \text{ errors in codeword}) = \begin{pmatrix} n \\ j \end{pmatrix} \cdot p^j \cdot (1-p)^{n-j},$$

$$P(\text{bit error} \mid j \text{ errors in a codeword}) = \frac{j}{n}.$$

Note that for the FIA the summation actually ranges from $t' + 1$ to n. For the theoretical plots, the FIA coding gain is positive for all SNR's above 3.2dB, and the BMA coding gain is positive for all SNR's above 11dB. Also, the FIA net coding gain is 1.5dB and 1.8dB for system BER's 10^{-5} and 10^{-8}, respectively, whereas the BMA net coding gain is small for both these system BER's.

To validate the simulation, decoding errors (those errors that give rise to undetected word errors) were disabled by giving the decoder knowledge of the transmitted codewords. Figures 3.2 and 3.3 compare the simulation results and the theoretical curves for the BMA and FIA, respectively. Each simulation curve coincides with its corresponding theoretical curve. Therefore, both simulation models are considered valid.

Next, both simulations were performed without restriction. Figures 3.4 and 3.5 compare the results with the theoretical curves and the Torrieri Bounds [8]. The Torrieri Bounds assume that the number of bit errors in a codeword has a binomial distribution (independent bit errors). If a codeword has t errors or less for the BMA or t' errors or less for the FIA, the codeword is assumed to be fully correctable; otherwise,

the decoder is assumed to add (worst case) t errors for the BMA or t' errors for the FIA to the codeword for the upperbound calculation, and subtract (best case) t errors for the BMA or t' errors for the FIA for the lowerbound calculation. For both the BMA and FIA simulations, the simulation curves were just above the theoretical curves, indicating that in the event of a decoding error, the decoder added errors to the received codeword slightly more often than it subtracted errors [8]. Figure 3.6 shows the simulation performance curves for both the BMA and the FIA. At a BER of 10^{-5}, the BMA coding gain was -.25dB, and the FIA coding gain was 1.4dB. The 10^{-8} BER coding gains are not available since the simulation was not performed for high SNR due to the exponential growth in required execution time.

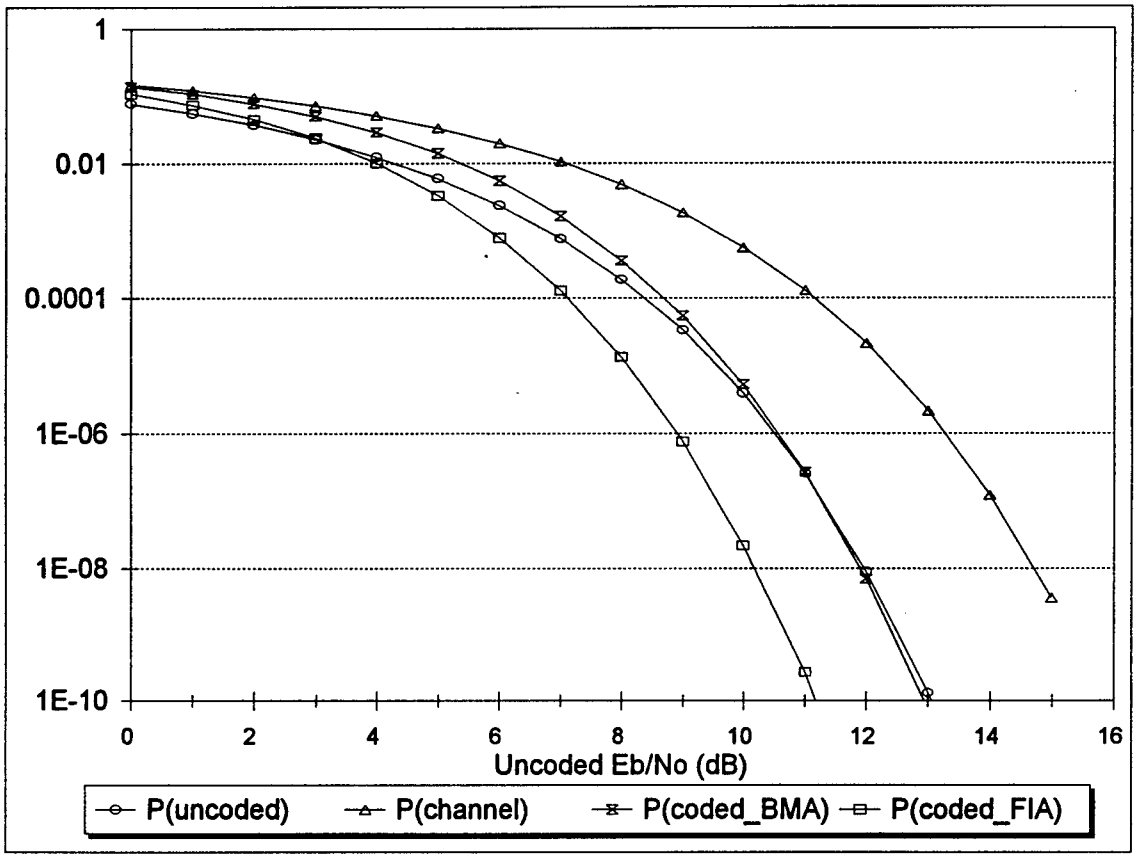


Figure 3.1. THEORETICAL PERFORMANCE CURVES FOR (17,9) BCH CODE

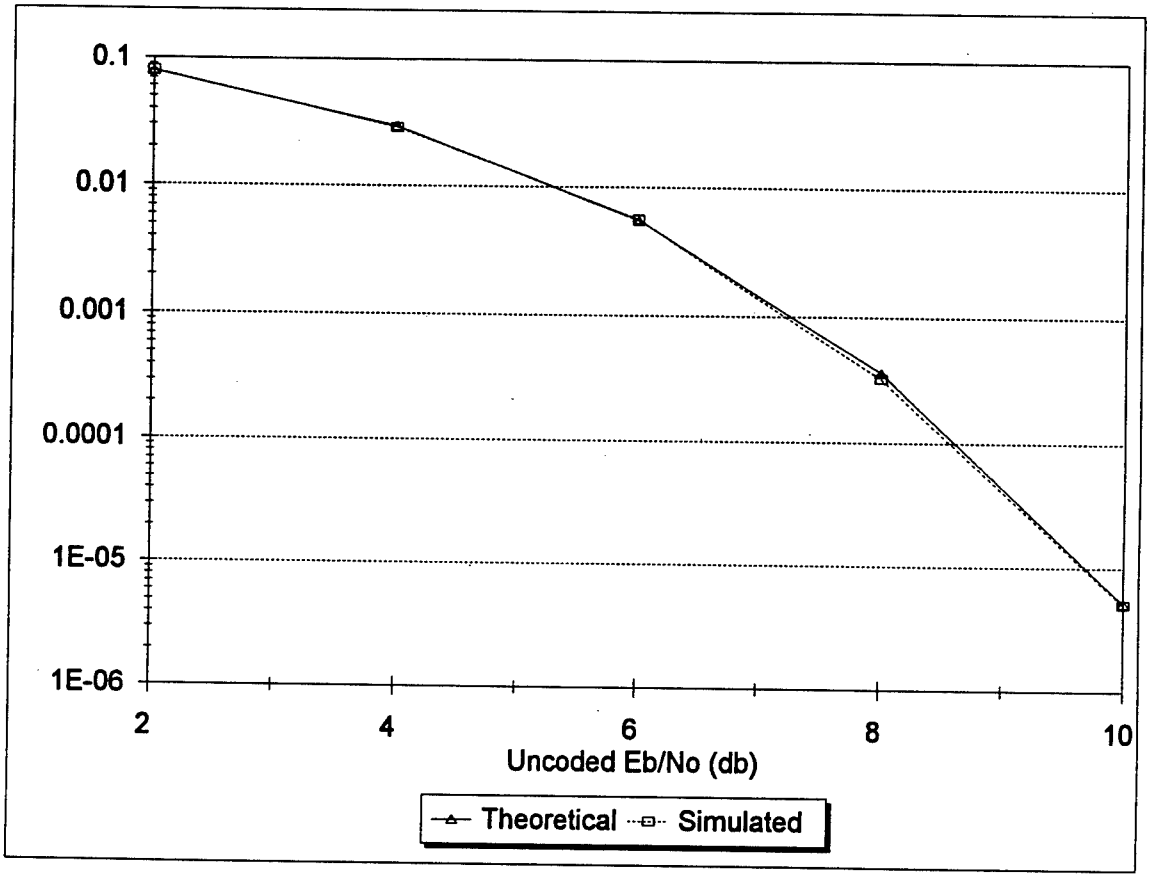


Figure 3.2. BMA STATISTICAL TEST WITH DECODING ERRORS DISABLED FOR (17,9) BCH CODE

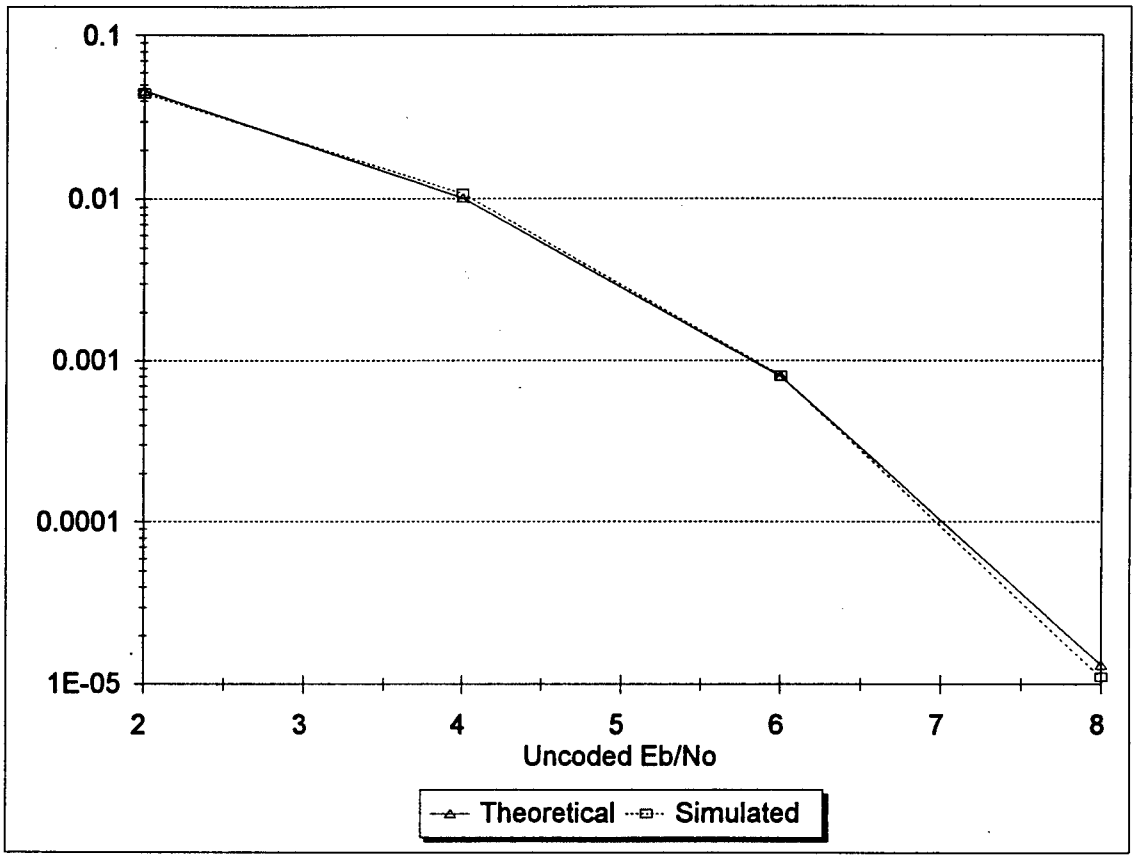


Figure 3.3. FIA STATISTICAL TEST WITH DECODING ERRORS DISABLED FOR (17,9) BCH CODE

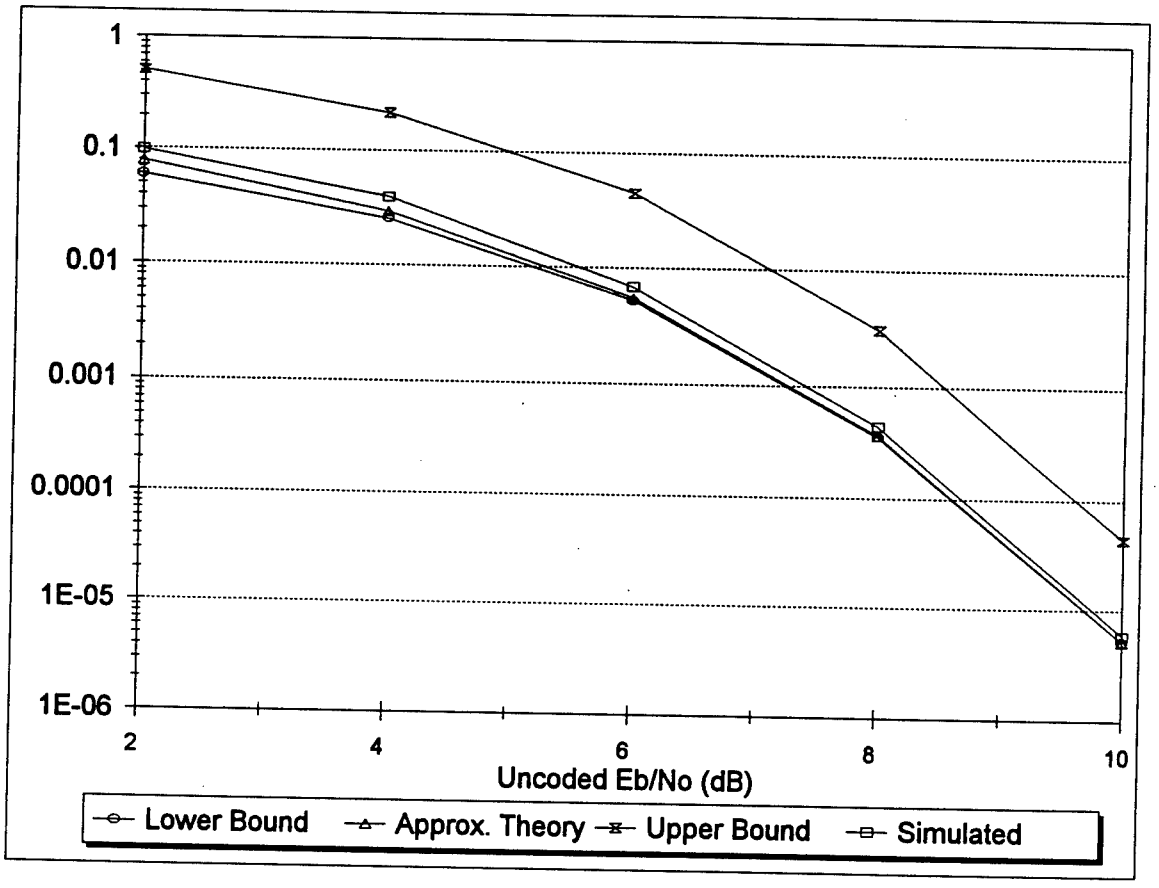


Figure 3.4. BMA STATISTICAL TEST WITH NO RESTRICTIONS FOR (17,9) BCH CODE

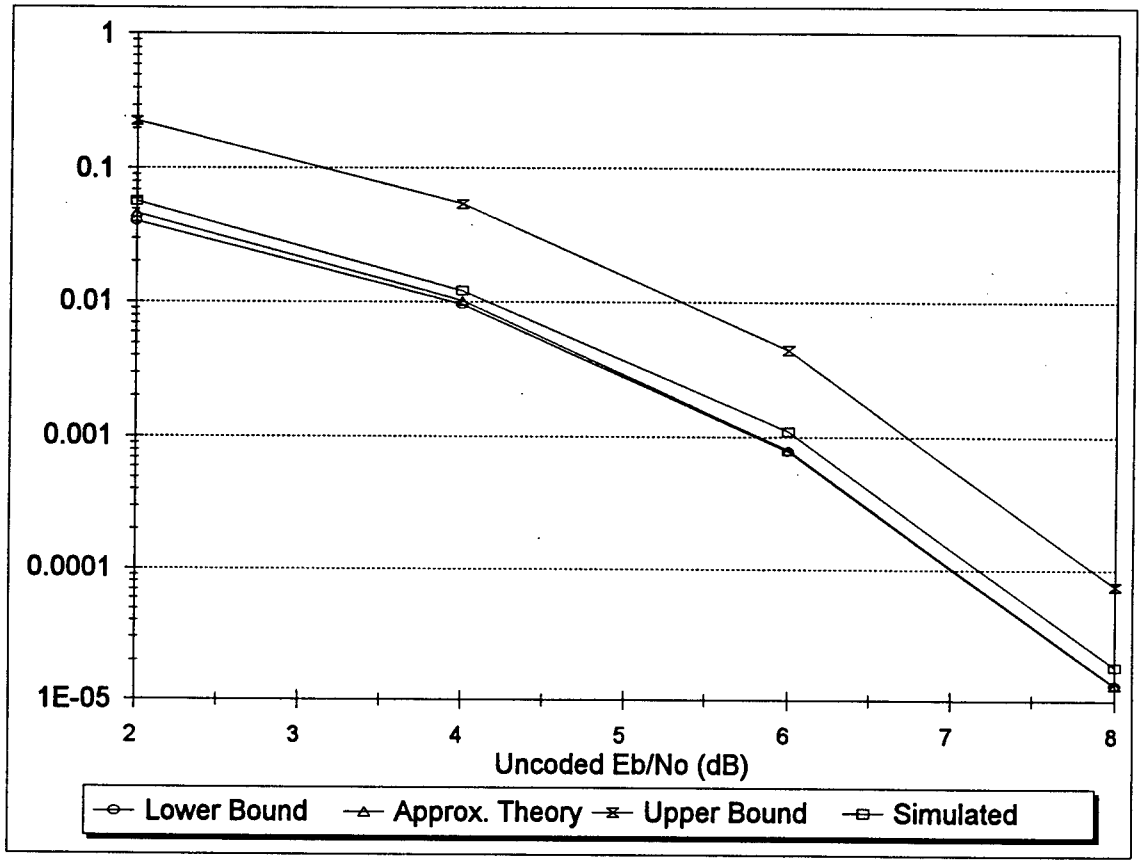


Figure 3.5. FIA STATISTICAL TEST WITH NO RESTRICTIONS FOR (17,9) BCH CODE

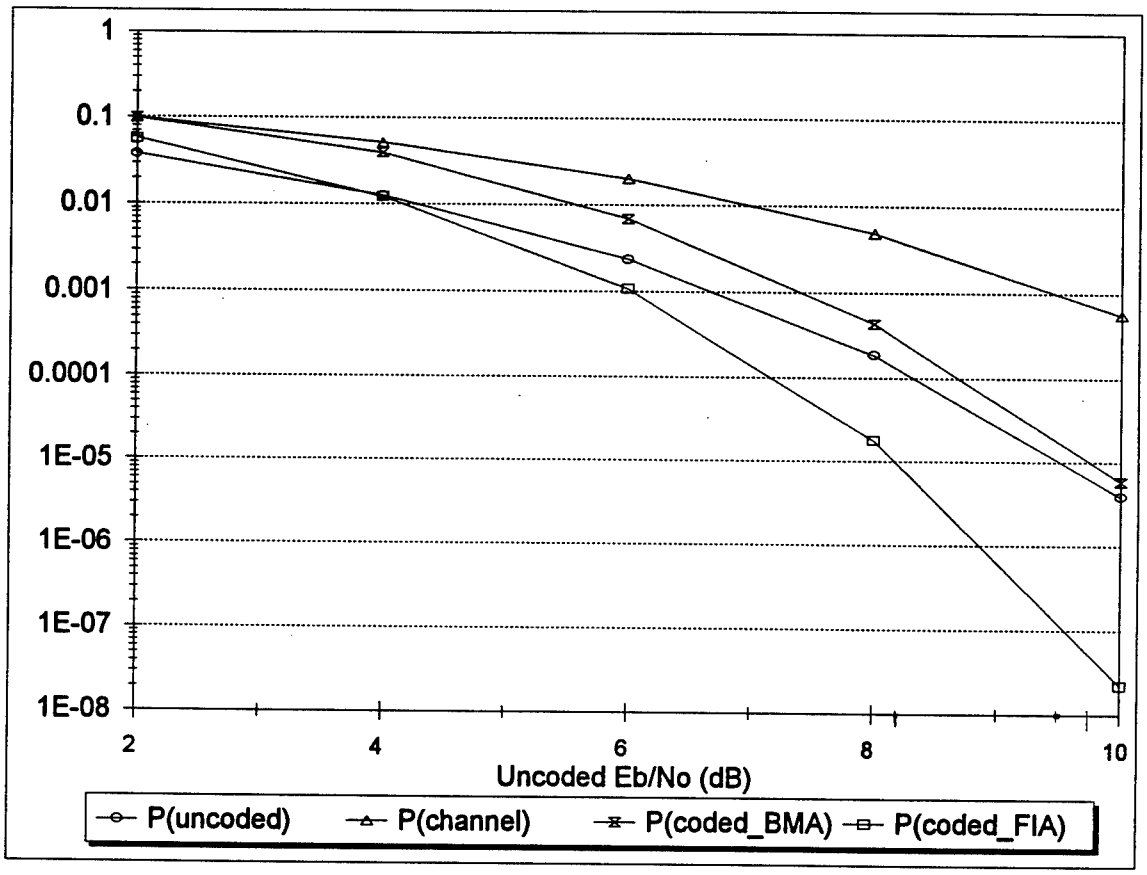


Figure 3.6. SIMULATION PERFORMANCE CURVES FOR (17,9) BCH CODE

CHAPTER IV

HARDWARE COMPARISONS

Two areas of comparison are examined in this chapter. First, the critical path is identified. The critical path is the hardware complexity of the algorithm bottleneck; it is measured with respect to the processing of one received bit. Second, hardware complexity of the entire algorithm is identified. Hardware complexity is similar to chip real-estate analysis but is viewed at a higher level. For this chapter, the complexity will be measured by the number of GF additions, GF multiplications, GF multiplicative inverses, arithmetic additions, arithmetic multiplications, comparisons, temporary memory slots, and permanent memory slots.

Critical Path for the BMA

Figure 4.1 shows the high-level hardware layout for the entire decoder. This layout calculates the syndrome S_1 using Horner's rule [1]. The latency of this syndrome calculation is the time required to perform $n-1$ GF multiplies and n GF additions. The maximum latency of the rest of the decoder is the time require to perform $n-1$ GF multiplies, 1 GF addition, 1 GF multiplicative inverse, $n-1$ arithmetic additions, and $2n$ comparisons. The calculation of syndromes is then the bottleneck, since $n-1$ GF additions will most likely require more time than 1 GF multiplicative

inverse, $n-1$ arithmetic additions, and $2n$ comparisons. Therefore, the critical path is 1 GF addition, 1 GF multiplication, and 1 memory element.

Critical Path for the FIA

Figure 4.2 shows the high-level hardware layout for the entire decoder. This layout also calculates the syndrome S_1 using Horner's rule [1]. The latency of the calculation and storage of syndromes is the time required to perform $(n+6)$ GF multiplies, n GF additions, plus RAM loading. The two syndrome RAM's must remain unchanged until the calculation of $\Omega(x)$, in the Forney Algorithm, is completed. In a worst case scenario, this storage time is approximately the time required to perform $6 \cdot n$ GF multiplies, $6 \cdot n$ GF additions, 3 GF reciprocals, 3 multiplies, n additions, and n compares, which is much longer than the latency time for the calculation and storage of syndromes, assuming that the time to load the two RAM's is relatively small. Therefore, the critical path lies between loading the RAM and completing the calculation of $\Omega(x)$. Since the critical path is measured with respect to the processing one received bit, the FIA has no obvious critical path, since the bottleneck cannot be dissected in this way.

As a side note, RAM 1 has six bytes of memory (8 bits/memory element) with a three bit addressing scheme, and RAM 2 has eight bytes of memory with a three bit addressing scheme. Both RAM's can be concatenated into one 14 byte RAM with a four bit addressing scheme, sacrificing access time.

Complexity Comparisons

Table 4.1 compares the hardware complexity of the two algorithms. Since the GF multiply is the most complex of all the operations, assuming polynomial representation for the field elements, the FIA is roughly 20 times more complex than the BMA.

Table 4.1. HARDWARE COMPLEXITY COMPARISON

Algorithm	GF +	GF x	GF Reciprocal	+	×	Comparison
BMA	2	2	1	1	0	3
FIA	34	42	3	5	1	10

Algorithm	Temporary Memory	Permanent Memory
BMA	37	0
FIA	128	14

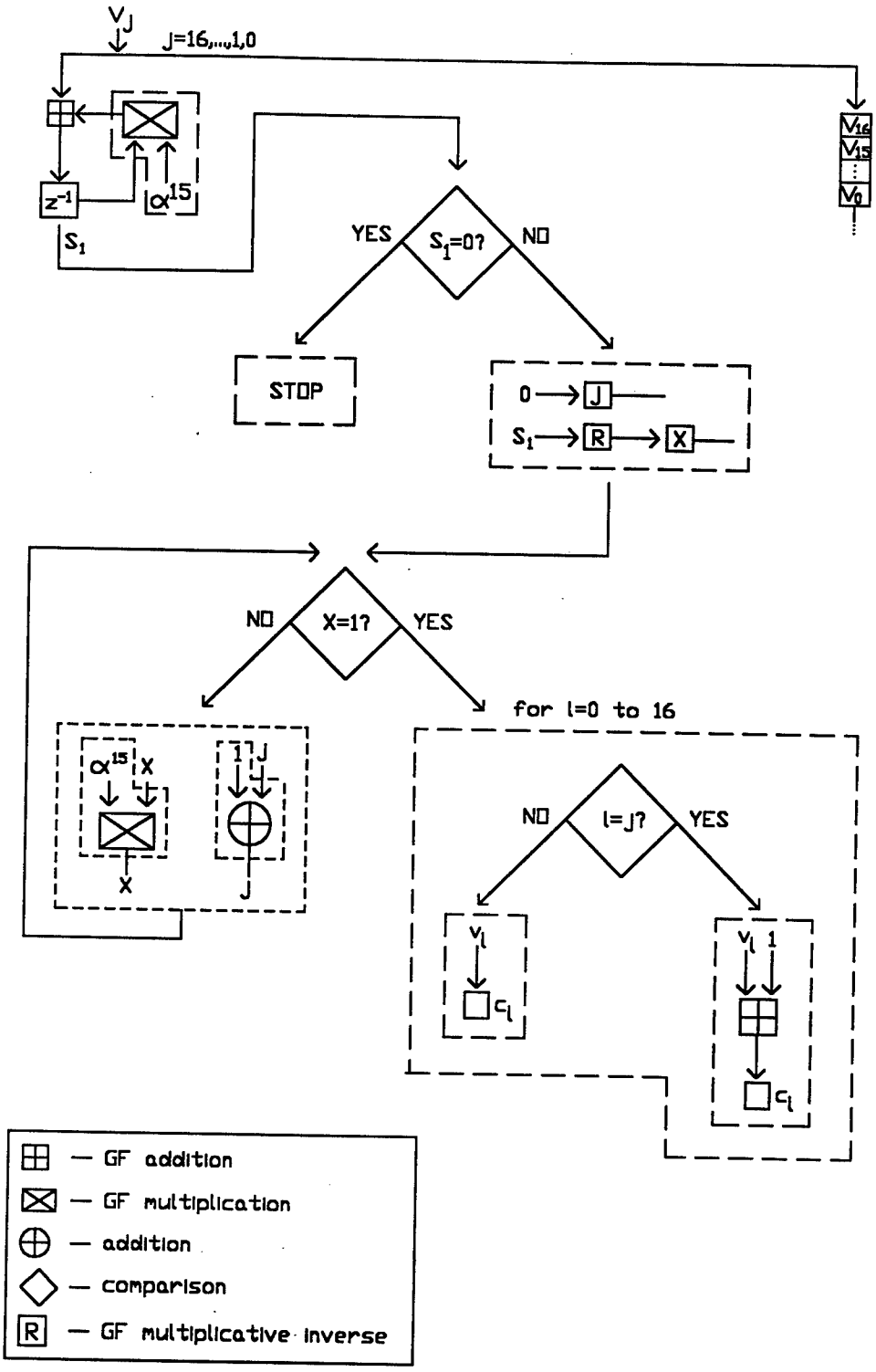


Figure 4.1. BMA HARDWARE LAYOUT FOR A (17,9) BCH DECODER

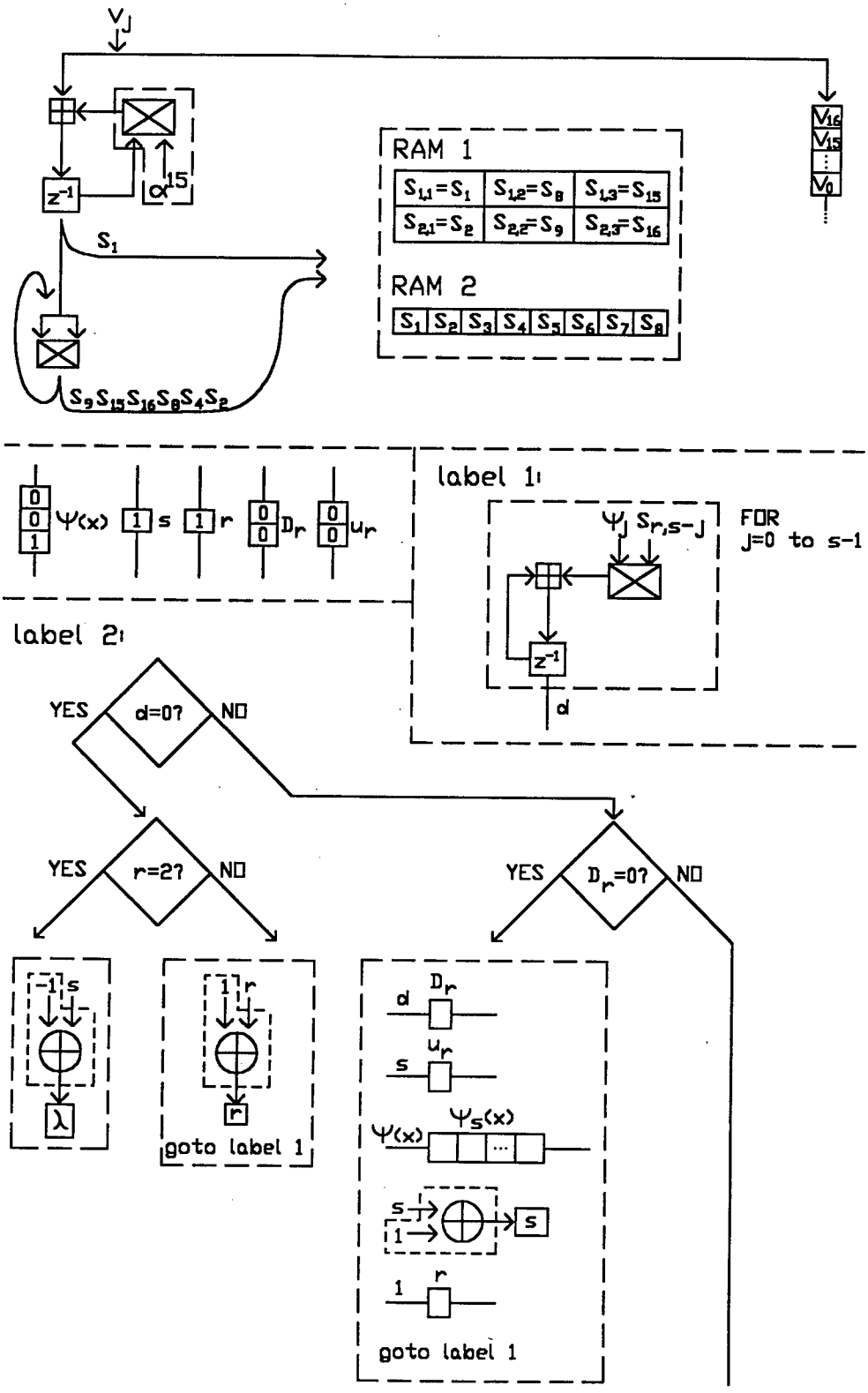


Figure 4.2. FIA HARDWARE LAYOUT FOR A (17,9) BCH DECODER

FIGURE 4.2. (continued)

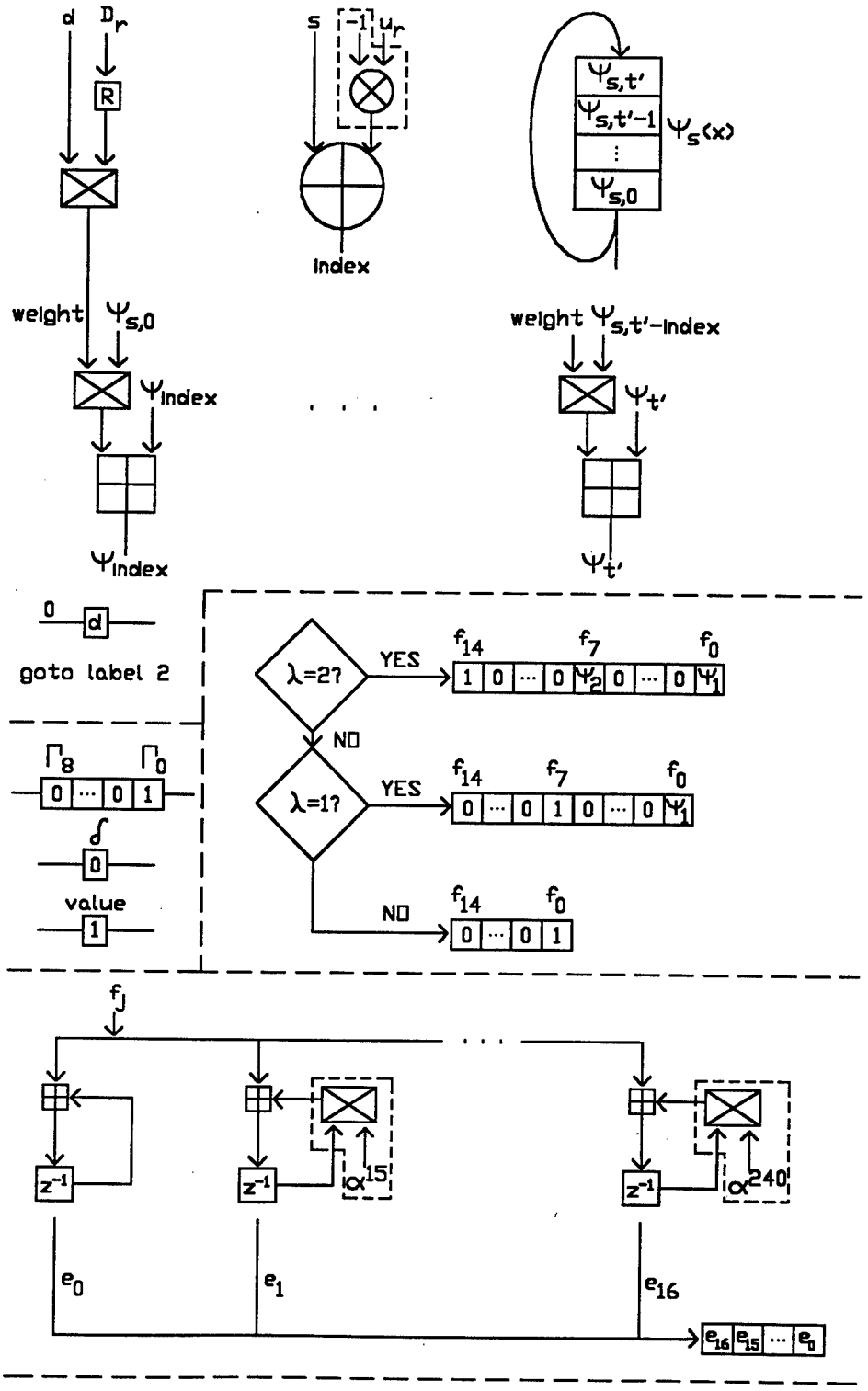


FIGURE 4.2. (continued)

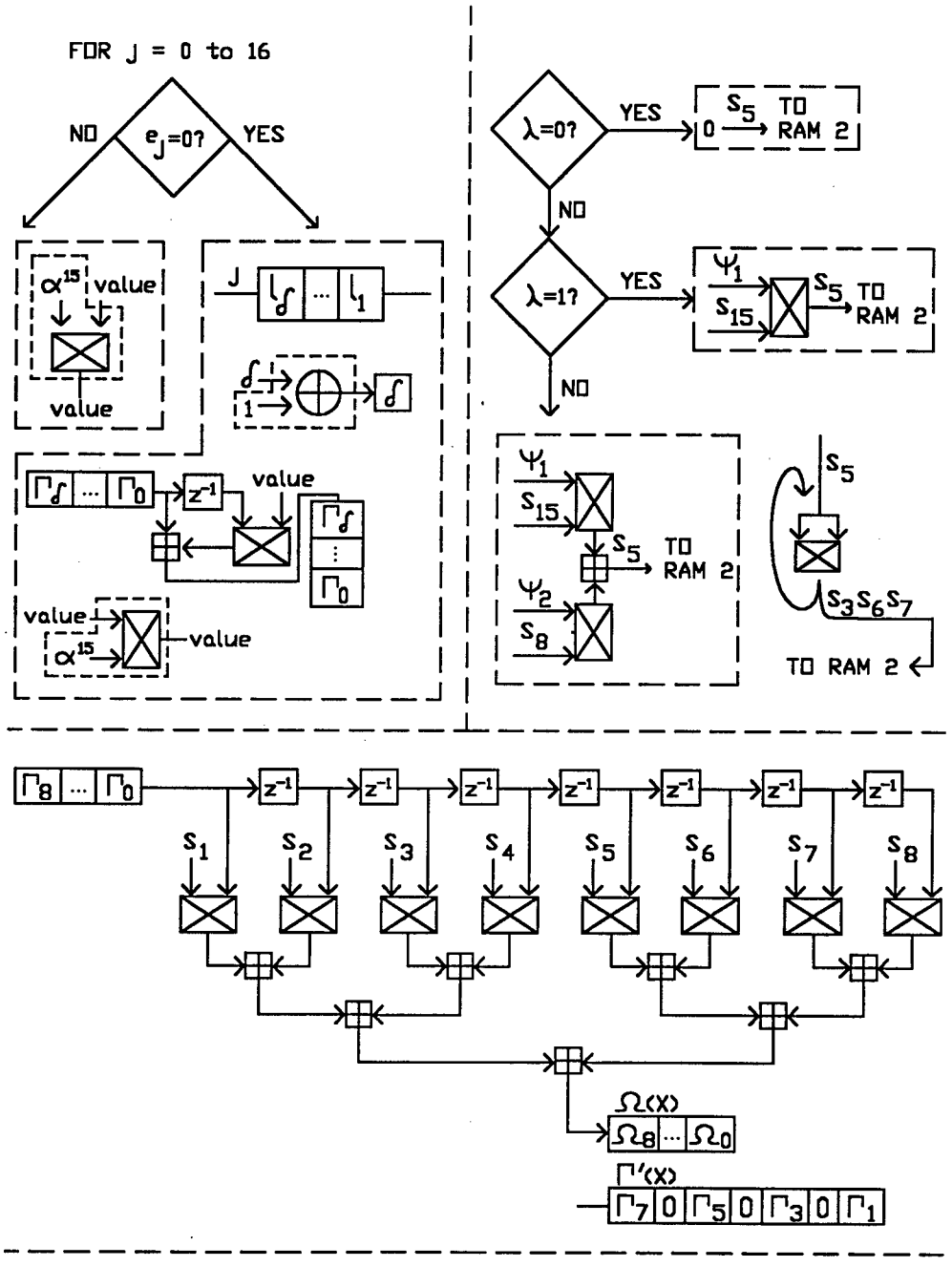
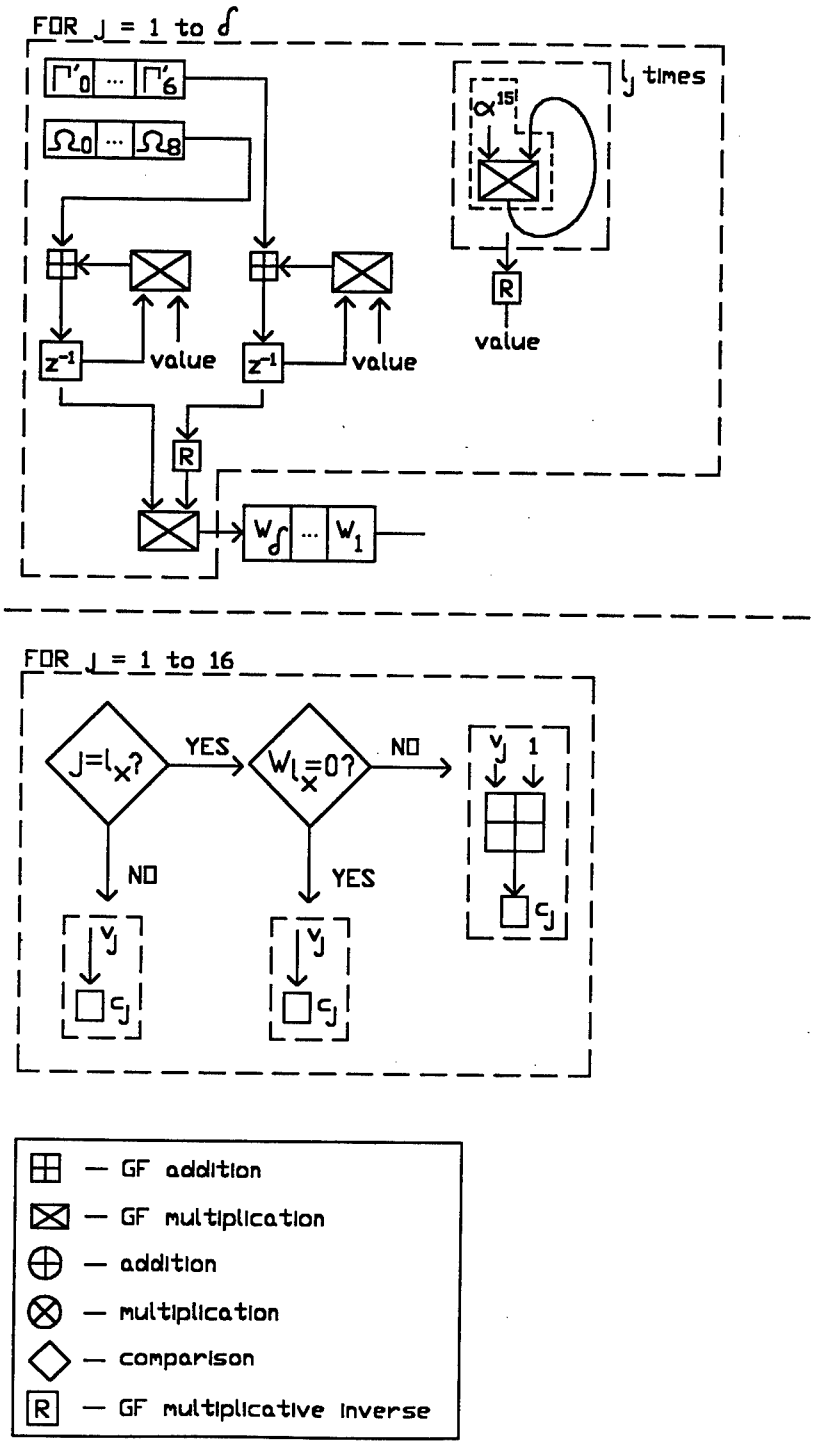


FIGURE 4.2. (continued)



CHAPTER V

SUMMARY AND CONCLUSIONS

The BMA uses only the designed distance of a BCH code, which is often smaller than the true minimum distance [1]; however, the FIA can be used to exploit some or all of the unused distance, in many cases increasing the number of correctable errors [3]. Although the number of correctable errors is an important performance indicator, a more complete indicator is the performance gain/hardware complexity ratio. At a BER of 10^{-5}, the performance gains are -.25dB and 1.4dB for the BMA and FIA, respectively; however, the hardware for the FIA decoder is roughly 20 times more complex than the hardware for the BMA. Also, the critical path for the BMA is 1 GF addition, 1 GF multiplication, and 1 memory element, whereas the FIA is so complex that the critical path not easily identified.

CHAPTER VI

FUTURE DIRECTIONS

Since the comparison of the BMA and FIA is code-specific, other codes with higher designed distances might be analyzed to give a more balanced comparison. Also, the Extended Fundamental Iterative Algorithm (EFIA) might be analyzed for net coding gain and hardware complexity [4].

BIBLIOGRAPHY

[1] Richard E. Blahut, *Theory and Practice of Error Control Codes,* Addison Wesley, 1984.

[2] William J. Ebel and Frank M. Ingels, "Confidence intervals for simulations using Reed-Solomon codes," Milcom, 1993.

[3] Gui-Liang Feng and Kenneth K. Tzeng, "Decoding cyclic and BCH codes up to actual minimum distance using nonrecurrent syndrome dependence relations," IEEE Trans. Information Theory, vol. 37, no. 6, pp. 1716-1722., Nov. 1991.

[4] Gui-Liang Feng and Kenneth K. Tzeng, "A new procedure for decoding cyclic and BCH codes up to actual minimum distance," IEEE Trans. Information Theory, vol. 40, no. 5, pp. 1364-1374., Sept. 1994.

[5] M. Jeruchim, "Techniques for estimating the bit error rate in the simulation of digital communication systems," IEEE JSAC, vol. SAC-2, pp. 153-171, Jan. 1984.

[6] Athanasios Papoulis, *Probability, Random Variables, and Stochastic Processes,* McGraw-Hill, 1991.

[7] Bernard Sklar, *Digital Communications Fundamentals and Applications*, Prentice Hall, 1988.

[8] D. J. Torrieri, "The Information-Bit Error Rate for Block Codes", IEEE Trans. on Communications, vol. Com-32, no. 4, April 1984.

[9] W. H. Tranter and K. L. Kosbar, "Simulation of communication systems," IEEE Communications Mag., pp. 26-35, July 1994.

[10] R. E. Ziemer and R. L. Peterson, *Digital Communications and Spread Spectrum Systems*, Macmillan, 1985.

[11] R. E. Ziemer and W. H. Tranter, *Principles of Communications*, Houghton Mifflin, 1990.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1996	Final report

4. TITLE AND SUBTITLE
A Hardware Analysis of the Fundamental Iterative Algorithm for Decoding A (17,9) Binary BCH Code

5. FUNDING NUMBERS

6. AUTHOR(S)
Roy L. Campbell, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Engineer Waterways Experiment Station
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER
Technical Report ITL-96-7

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Corps of Engineers
Washington, DC 20314-1000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Available from National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Berlekamp-Massey Algorithm (BMA) is commonly used in BCH decoding, but the Fundamental Iterative Algorithm (FIA) can, in many instances, correct more errors. The trade-off lies in hardware complexity. The BMA has concise stages that do not require addressable memory, whereas the FIA struggles with memory management and complex stages. For the (17,9) BCH code over GF(256) with $g(x)=m_1(x)$, the FIA can correct up to two errors, but the BMA can correct only one error. Also, for a BER of 10^{-5}, the coding gains for the BMA and FIA are -.25dB and 1.4dB, respectively, which makes the FIA seem to be the better algorithm. The main drawback is the FIA is approximately 20 times more complex than the BMA. Also, the FIA does not have a critical path that is easily identified.

14. SUBJECT TERMS
Berlekamp-Massey Algorithm
Fundamental Iterative Algorithm

15. NUMBER OF PAGES
44

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED		